Міністерство освіти і науки України
Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №3
З дисципліни «Технології розроблення програмного забезпечення»
Тема: **«Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.»**
Powershell terminal

Виконав:
Студент групи ІА-22
Парій І. Р.

Перевірив:
Мягкий М. Ю.

Київ-2024

Тема:
Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.

Мета:
Засвоїти основні типи діаграм, які використовуються для моделювання програмних систем, зокрема діаграми розгортання, компонентів, взаємодій і послідовностей. Навчитися будувати й аналізувати такі діаграми, а також застосовувати їх для опису архітектури та процесів у програмних системах.

Хід роботи

1. Діаграма розгортання

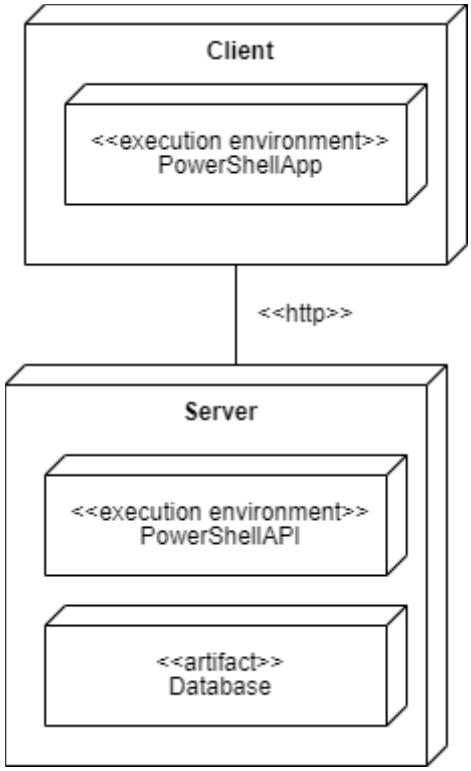

Рис. 1 — Діаграма розгортання

Діаграма демонструє розподіл компонентів проєкту між клієнтом і сервером. Основні елементи включають:

1. Клієнт:
    - PowerShellApp (Execution Environment): Це середовище, у якому працює PowerShell-термінал. Воно відповідає за відображення користувацького інтерфейсу терміналу та забезпечує можливість:
        - Налаштування кольорів синтаксичних конструкцій.

- Зміну розмірів та фону вікна.
- Запуск PowerShell-команд та виконуваних файлів.
- Підтримку роботи в кількох вікнах або вкладках для одночасного виконання різних команд.

2. Сервер:
   - PowerShellAPI (Execution Environment): Це середовище забезпечує серверну логіку для обробки команд та запитів від клієнта. API може взаємодіяти з командним інтерпретатором PowerShell та передавати результати виконання команд назад на клієнт.
   - Database (Artifact): База даних слугує для збереження конфігурацій користувача (наприклад, налаштувань кольорів, розмірів вікна, попередніх команд) або логів виконання команд для подальшого аналізу.

Інтеракція між компонентами: Клієнтська програма PowerShellApp надсилає запити до серверного компонента PowerShellAPI для виконання команд або отримання даних. Серверна частина взаємодіє з базою даних для збереження/читання даних і повертає результати клієнту для відображення в терміналі.

Ця архітектура забезпечує можливість налаштування середовища та гнучке управління виконанням команд, підтримуючи масштабованість та інтеграцію з іншими системами.

2. Діаграма компонентів

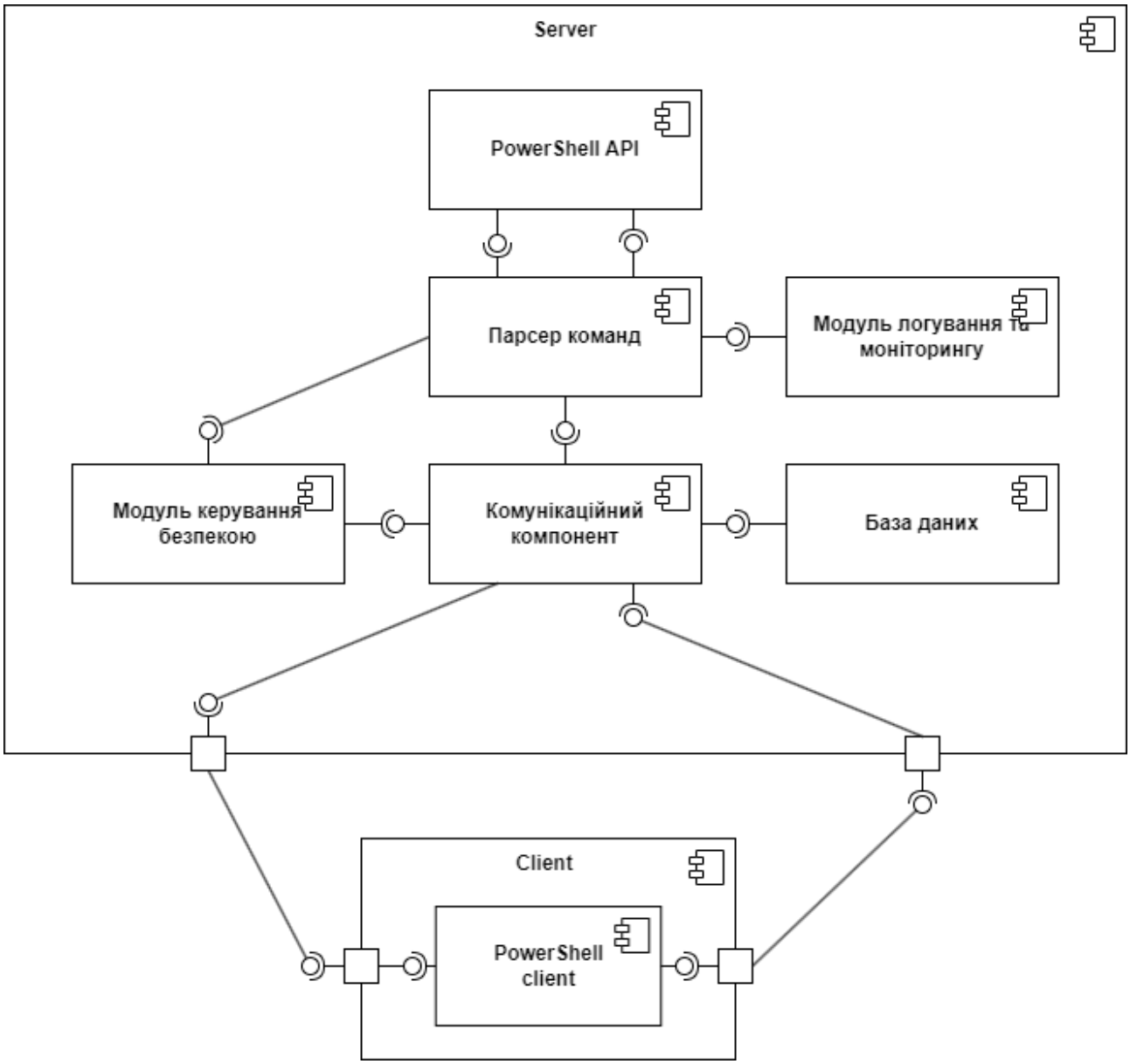

Рис. 2 — Діаграма компонентів

Діаграма компонентів ілюструє архітектуру системи "PowerShell terminal", що забезпечує взаємодію клієнт-сервер і підтримує функціональність налаштовуваного термінала PowerShell. Нижче подано опис ключових компонентів:

1. Сервер:
   - PowerShell API: Забезпечує доступ до функціоналу PowerShell і дозволяє виконувати команди та скрипти, а також взаємодіяти з операційною системою.
   - Парсер команд: Обробляє введені команди, інтерпретує їх і передає у PowerShell API для виконання.

- Модуль логування та моніторингу: Відповідає за запис історії виконаних команд, системних подій та моніторинг активності для забезпечення прозорості та діагностики.
- База даних: Зберігає інформацію про виконані команди, історію сесій, налаштування користувача (наприклад, кольори, розмір вікна тощо).
- Модуль керування безпекою: Забезпечує перевірку прав доступу, автентифікацію користувачів та шифрування даних для безпечної роботи.
- Комунікаційний компонент: Служить посередником між клієнтом і сервером, забезпечуючи обмін даними через стандартизований протокол.

2. Клієнт:
   - PowerShell Client: Це графічний інтерфейс користувача, який надає можливість:
     - Виконувати команди PowerShell;
     - Налаштовувати кольори синтаксису, фон, розмір і вигляд вікна;
     - Працювати в багатовіконному режимі (вкладки або розділення вікна).

   Компоненти клієнта і сервера взаємодіють через Комунікаційний компонент, який забезпечує передачу команд від клієнта до сервера та повернення результатів їх виконання.

Взаємодія:
Клієнт відправляє запити на сервер, сервер обробляє їх через відповідні компоненти (Парсер команд, API, базу даних тощо) та повертає результат. Завдяки цій модульній архітектурі система залишається гнучкою, розширюваною та зручною у використанні.

3. Діаграма послідовностей

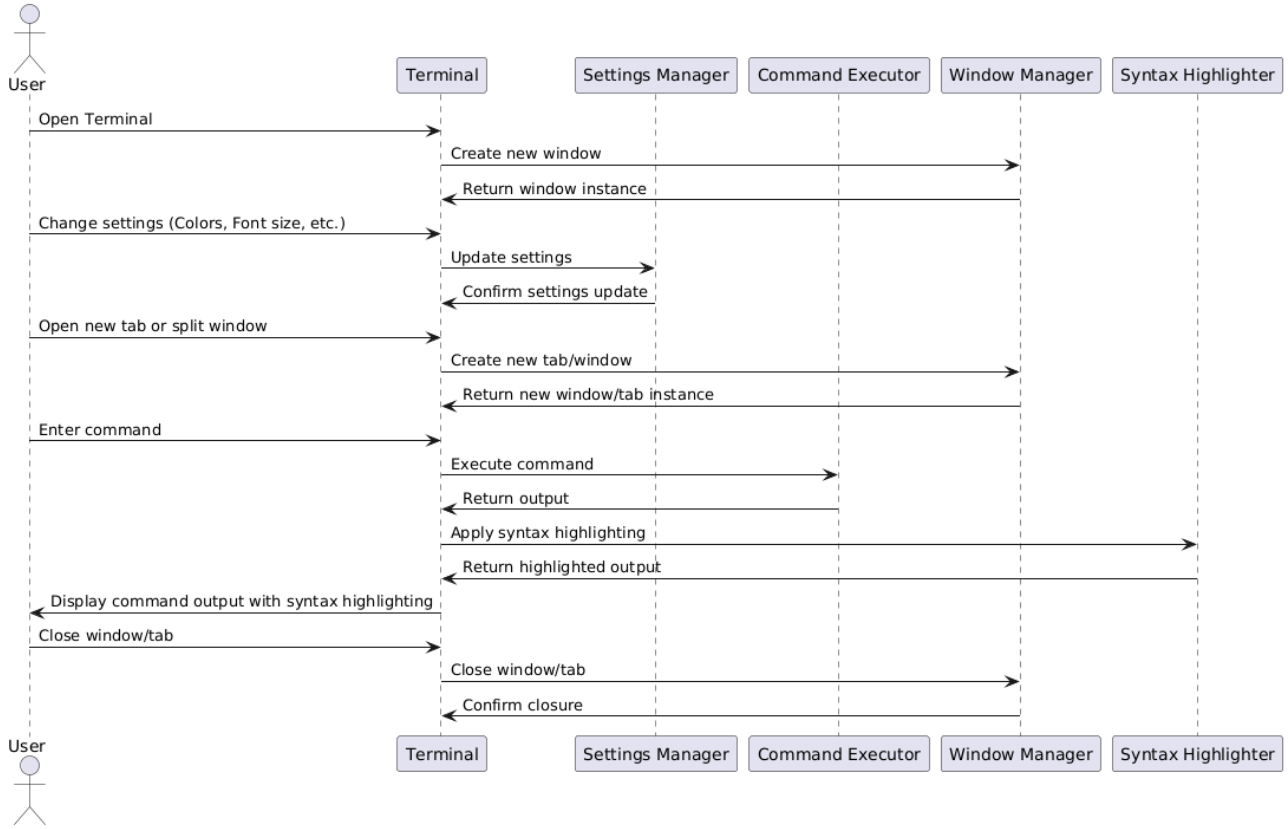

Рис. 3 — Діаграма послідовностей

Діаграма послідовностей описує основний сценарій роботи користувача з терміналом, розробленим для PowerShell, з урахуванням можливостей налаштування та багатовіконного режиму.

1. Відкриття терміналу:
   - Користувач відкриває термінал.
   - Створюється нове вікно, що ініціюється компонентом Terminal.
   - Після створення нового вікна повертається відповідний екземпляр.
2. Налаштування параметрів:
   - Користувач змінює налаштування, наприклад, кольори синтаксису, шрифт чи фон.
   - Компонент Settings Manager оновлює ці налаштування та підтверджує їхнє застосування.
3. Відкриття нової вкладки або поділу вікна:
   - Користувач створює нову вкладку або розділяє вікно для паралельної роботи.
   - Компонент Window Manager виконує відповідну дію та повертає екземпляр нової вкладки чи розділеного вікна.

4. Виконання команди:
    - Користувач вводить команду PowerShell.
    - Команда виконується через Command Executor, який повертає результат.
    - Результат передається в Syntax Highlighter, який додає підсвічування синтаксису.
    - Підсвічений результат відображається в інтерфейсі.
5. Закриття вікна/вкладки:
    - Користувач закриває вкладку чи вікно.
    - Компонент Terminal ініціює процес закриття, який завершується підтвердженням.

Дана діаграма чітко структурує взаємодію компонентів терміналу, забезпечуючи їхню узгоджену роботу для задоволення вимог користувача.

Висновки:

Отримали знання про основні види діаграм, що використовуються для моделювання програмних систем, зокрема розгортання, компонентів, взаємодій і послідовностей. Освоїли навички створення й аналізу цих діаграм та їхнє практичне застосування для опису архітектурних рішень і процесів у програмних системах.